# Aplicaciones Web

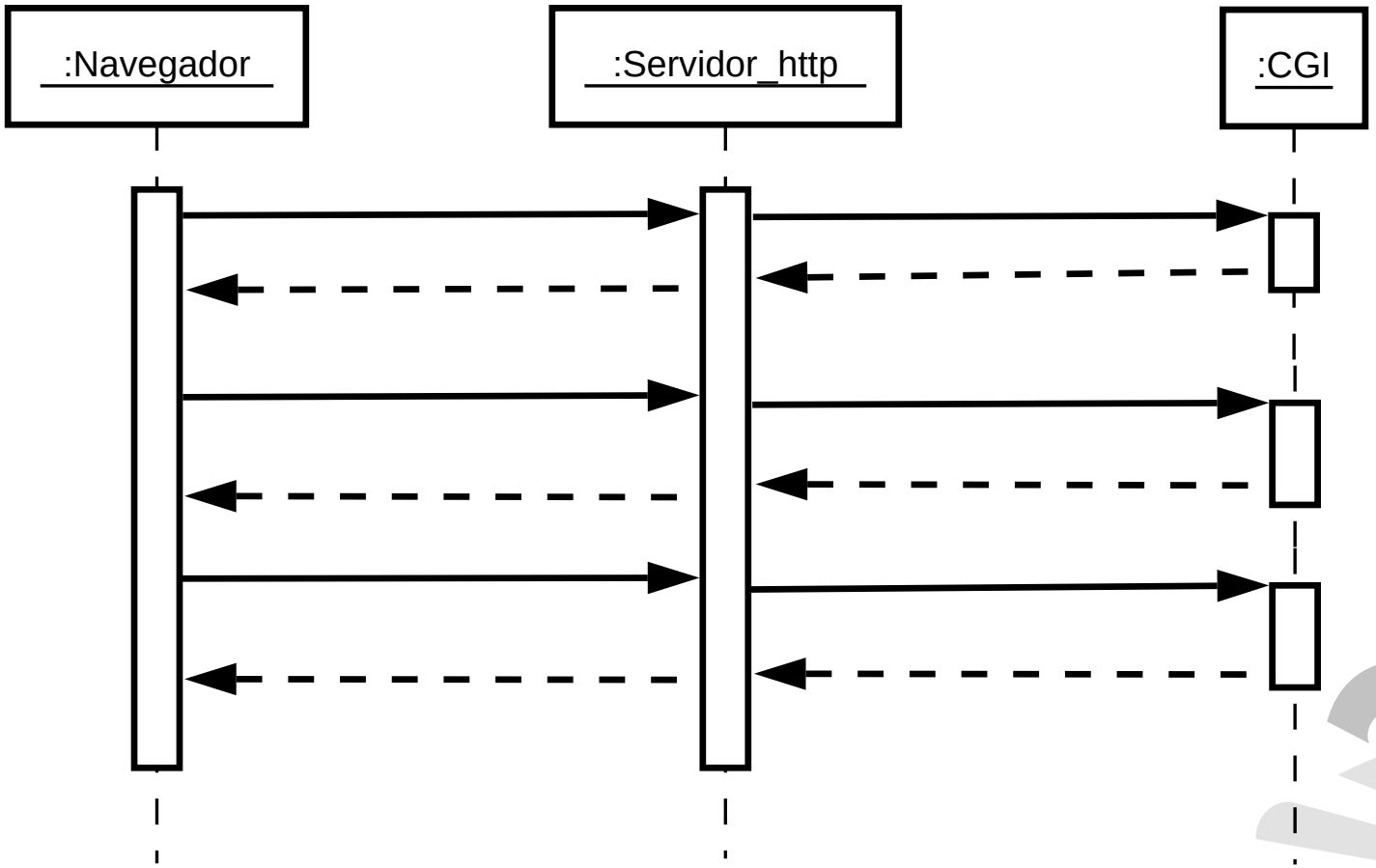
## Más alla de CGI

David Cabrero Souto

Grupo MADS (http://www.grupomads.org/)
Universidade da Coruña

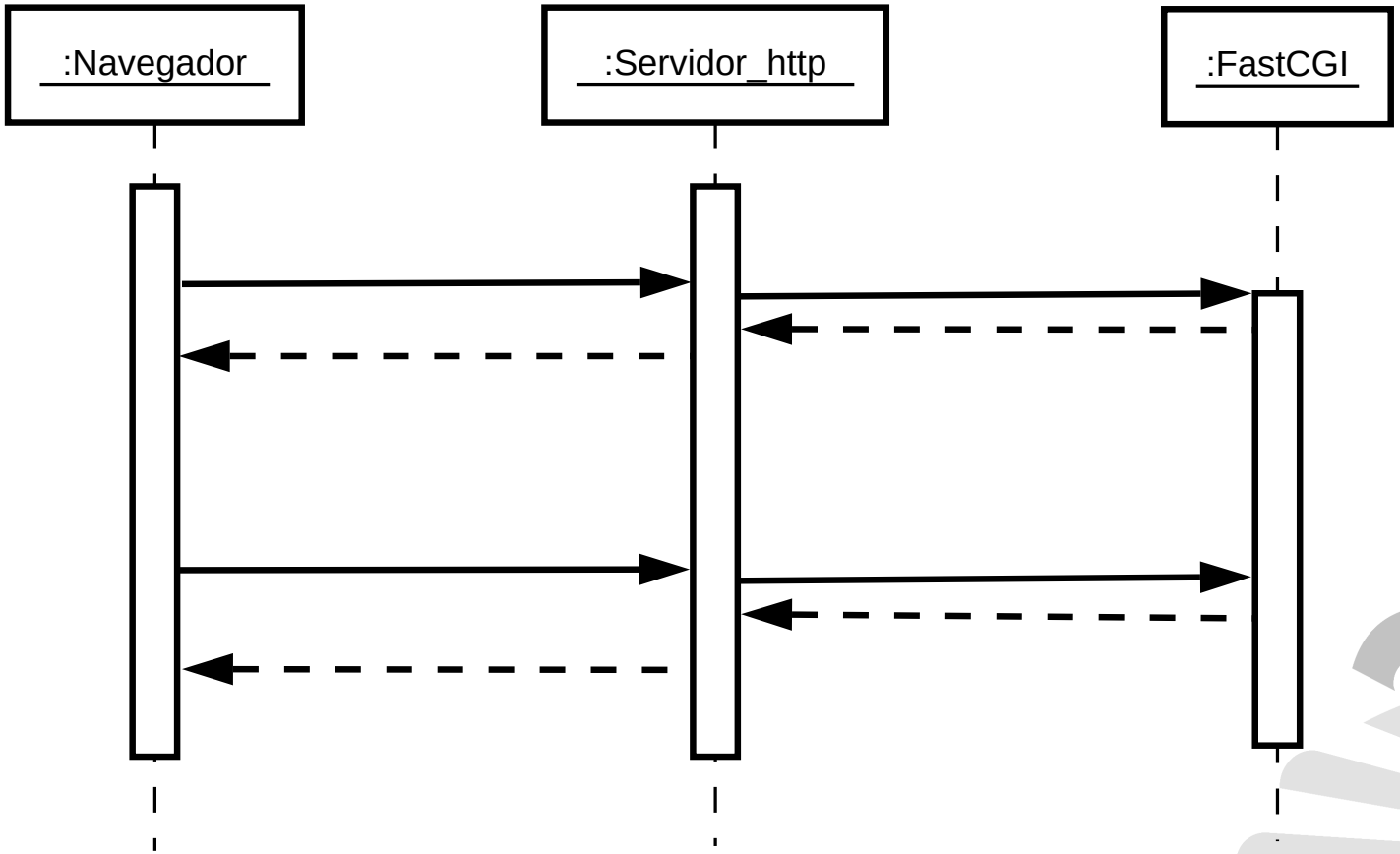
# Problema de CGI

- Problema de CGI: coste de lanzar un proceso por petición.



# Fast CGI

- El proceso se lanza con la primera petición.
- No termina, continúa ejecutándose.



# Extensiones del servidor

- Módulos Apache para lenguajes: python, php, perl, ruby, ...
- El intérprete “forma parte” del servidor.
- Adicionalmente se ofrece acceso a características propias del servidor.

```
form mod_python import apache
...
 apache.log_error("Movimiento incorrecto")
```

# Extensiones del servidor

- Módulos Apache para lenguajes: python, php, perl, ruby, . . .
- El intérprete “forma parte” del servidor.
- Adicionalmente se ofrece acceso a características propias del servidor.

```
form mod_python import apache
...
 apache.log_error("Movimiento incorrecto")
```

# Lenguaje incrustado (II)

- El código de la aplicación se introduce en la página.
- Ejemplo python PSP:

```
<html>
...
<%
if form.has_key('name'):
   greet = 'Hello, %s!' % form['name'].capitalize()
else:
   greet = 'Hello there!'
# end
%>
  <h1><%= greet %></h1>
...
</html>
```

# Lenguaje incrustado (II)

- El código de la aplicación se introduce en la página.
- Ejemplo python PSP:

```
<html>
...
<%
if form.has_key('name'):
   greet = 'Hello, %s!' % form['name'].capitalize()
else:
   greet = 'Hello there!'
# end
%>
  <h1><%= greet %></h1>
...
</html>
```

# Lenguaje incrustado (II)

- Ejemplo PHP:

```
<html>

<head>
<title>Example #1 TDavid's Very First PHP Script ever
</head>

<body>
  <? print(Date("l F d, Y")); ?>
</body>
</html>
```

- Peligro: Olvidar MVC.
- Peligro: Las páginas dejan de ser HTML.

# Lenguaje incrustado (II)

- Ejemplo PHP:

```
<html>

<head>
<title>Example #1 TDavid's Very First PHP Script ever
</head>

<body>
  <? print(Date("l F d, Y")); ?>
</body>
</html>
```

- Peligro: Olvidar MVC.
- Peligro: Las páginas dejan de ser HTML.

# Lenguaje incrustado (II)

- Ejemplo PHP:

```
<html>

<head>
<title>Example #1 TDavid's Very First PHP Script ever
</head>

<body>
  <? print(Date("l F d, Y")); ?>
</body>
</html>
```

- Peligro: Olvidar MVC.
- Peligro: Las páginas dejan de ser HTML.

# Contenedores de aplicaciones

- Ejemplo: Java.
  - Servlet = clase java.
  - Contendor de servlets: mapea urls a servlets.
  - Contendor de aplicaciones: añade seguridad, concurrencia, transacciones, ...

# Contenedores de aplicaciones

- Ejemplo: Java.
  - Servlet = clase java.
  - Contendor de servlets: mapea urls a servlets.
  - Contendor de aplicaciones: añade seguridad, concurrencia, transacciones, ...

# Autenticación

- Autenticación HTTP.
  - Basada en login/password.
  - La información viaja en las cabeceras.
  - Podemos recuperar la información desde nuestra aplicación.

```
def authenhandler(req):
    pw = req.get_basic_auth_pw()
    user = req.connection.user
```

  - Con `AuthType Basic` la clave viaja en claro.
  - Con `AuthType Digest` se manda un resumen MD5.
- Autenticación “ad-hoc”.
  - Login y password son argumentos de la petición.
- En ambos casos: HTTPS.

# Autenticación

- Autenticación HTTP.
  - Basada en login/password.
  - La información viaja en las cabeceras.
  - Podemos recuperar la información desde nuestra aplicación.

```
def authenhandler(req):
    pw = req.get_basic_auth_pw()
    user = req.connection.user
```

  - Con `AuthType Basic` la clave viaja en claro.
  - Con `AuthType Digest` se manda un resumen MD5.
- Autenticación “ad-hoc”.
  - Login y password son argumentos de la petición.
- En ambos casos: HTTPS.

# Autenticación

- Autenticación HTTP.
  - Basada en login/password.
  - La información viaja en las cabeceras.
  - Podemos recuperar la información desde nuestra aplicación.

```
def authenhandler(req):
    pw = req.get_basic_auth_pw()
    user = req.connection.user
```

  - Con `AuthType Basic` la clave viaja en claro.
  - Con `AuthType Digest` se manda un resumen MD5.
- Autenticación “ad-hoc”.
  - Login y password son argumentos de la petición.
- En ambos casos: HTTPS.

# Autenticación

- Autenticación HTTP.
  - Basada en login/password.
  - La información viaja en las cabeceras.
  - Podemos recuperar la información desde nuestra aplicación.

```
def authenhandler(req):
    pw = req.get_basic_auth_pw()
    user = req.connection.user
```

  - Con `AuthType Basic` la clave viaja en claro.
  - Con `AuthType Digest` se manda un resumen MD5.
- Autenticación “ad-hoc”.
  - Login y password son argumentos de la petición.
- En ambos casos: HTTPS.

# Autenticación

- Autenticación HTTP.
  - Basada en login/password.
  - La información viaja en las cabeceras.
  - Podemos recuperar la información desde nuestra aplicación.

```
def authenhandler(req):
    pw = req.get_basic_auth_pw()
    user = req.connection.user
```

  - Con `AuthType Basic` la clave viaja en claro.
  - Con `AuthType Digest` se manda un resumen MD5.
- Autenticación “ad-hoc”.
  - Login y password son argumentos de la petición.
- En ambos casos: HTTPS.

# Autenticación

- Autenticación HTTP.
  - Basada en login/password.
  - La información viaja en las cabeceras.
  - Podemos recuperar la información desde nuestra aplicación.

```
def authenhandler(req):
    pw = req.get_basic_auth_pw()
    user = req.connection.user
```

  - Con `AuthType Basic` la clave viaja en claro.
  - Con `AuthType Digest` se manda un resumen MD5.
- Autenticación “ad-hoc”.
  - Login y password son argumentos de la petición.
- En ambos casos: HTTPS.

# Autenticación

- Autenticación HTTP.
  - Basada en login/password.
  - La información viaja en las cabeceras.
  - Podemos recuperar la información desde nuestra aplicación.

```
def authenhandler(req):
    pw = req.get_basic_auth_pw()
    user = req.connection.user
```

  - Con `AuthType Basic` la clave viaja en claro.
  - Con `AuthType Digest` se manda un resumen MD5.
- Autenticación “ad-hoc”.
  - Login y password son argumentos de la petición.
- En ambos casos: HTTPS.

# Autenticación

- Autenticación HTTP.
  - Basada en login/password.
  - La información viaja en las cabeceras.
  - Podemos recuperar la información desde nuestra aplicación.

```
def authenhandler(req):
    pw = req.get_basic_auth_pw()
    user = req.connection.user
```

  - Con `AuthType Basic` la clave viaja en claro.
  - Con `AuthType Digest` se manda un resumen MD5.
- Autenticación “ad-hoc”.
  - Login y password son argumentos de la petición.
- En ambos casos: HTTPS.

# No usar servidor http

- Protocolo HTTP tiene varias partes: MUST, SHOULD, MAY.
- Tiene sentido en: embebidos, aplicaciones especiales.

# No usar servidor http

- Protocolo HTTP tiene varias partes: MUST, SHOULD, MAY.
- Tiene sentido en: embebidos, aplicaciones especiales.